Synthetic News Radio

by K. Emnett
C. Schmandt

This paper describes a system that uses speech recognition and clustered text news stories to automatically find story boundaries in an audio news broadcast and provides a semantic representation that can match audio news stories of similar content. This system creates a personal, synthetic newscast by extracting stories, based on user interests, from multiple hourly newscasts and then reassembling them into a single recording at the end of the day. Interaction is via graphical and telephone-based interfaces, with newscasts delivered over a local area network or to wireless audio pagers.

Use of personalized on-line news services is increasing, and, of course, numerous search engines are available on the Web. But such services face limitations in being centered on text news sources and employing screen-based user interfaces. Although these limitations are quite acceptable when working with a traditional computer, they prevent access in a variety of situations, such as while out of the office or driving. At a deeper level, we anticipate that a significant portion of the population is never going to want to read news on a computer screen.

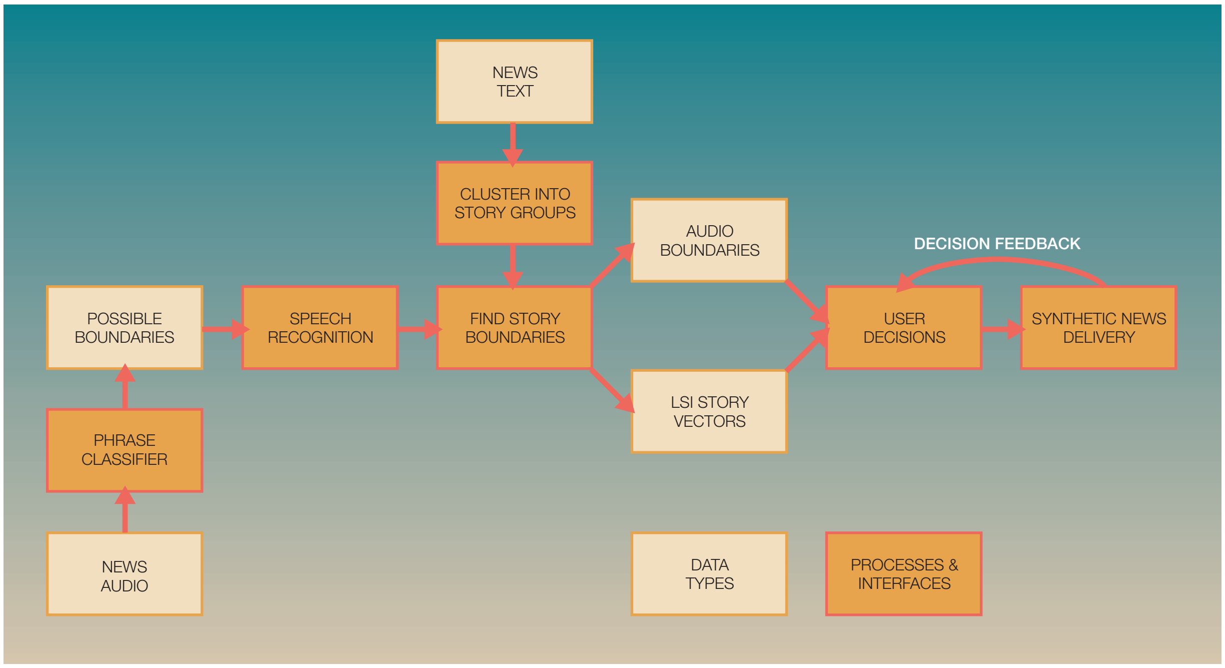
Synthetic News Radio (SNR) provides a different way of accessing and personalizing the news. It deals with radio newscasts and supports several audio-only user interfaces, in addition to a graphical user interface (GUI). The SNR user does not make queries, but rather, while listening to the news, can navigate between stories and ask to have particular stories tracked or ignored. Later in the day, SNR generates synthetic newscasts, updated hourly, reflecting the interests previously expressed by the user. As any synthetic newscast may contain new material, the user can similarly navigate and express preferences while listening to it.

SNR is somewhat different from previous attempts to browse audio news recordings[1–3] in that the listener need not use text to express interest or disinterest in story lines; he or she simply listens to the audio and reacts while hearing any story. Rather than searching archives of past news, we wish to generate filters to allow listeners to sift through ongoing stories. Although SNR utilizes an underlying textual representation and employs text-based information retrieval techniques, this mechanism is hidden from the user.

SNR uses commercially available speech recognition that has been designed for automatic speech-to-text dictation. Providing a transcript of a newscast, with an unknown speaker or several speakers, is not what these recognizers have been developed for; however, there are a number of similar and inexpensive such products on the market. As discussed in this paper, dictation recognizers do not perform particularly well at information retrieval tasks, and more appropriate speech recognition techniques are under development in research laboratories. It was our intent in this project to demonstrate that with other sources

0018-8670/00/$5.00 © 2000 IBM

Figure 1 Block diagram of the system



of knowledge, even poor transcription could satisfy queries, and, more important, to provide a realistic application for which we could explore appropriate user interfaces. Other researchers have also shown success in using additional sources of information, such as speaker identification, along with a transcript.[4]

System architecture

Output from one of the commercial speech recognizers is not enough to automatically determine story boundaries. We approach this problem by building a semantic representation of news stories from a corpus of current news received on a separate text feed. These stories are related to what is heard on the hourly news, but neither represents a concise summary nor provides a one-to-one mapping to the newscast in any particular hour. Analysis of this text feed does lead to an underlying semantic representation, updated hourly, of today's top stories; this representation is used in concert with speech recognition and phrase boundary determination techniques to segment and classify the audio news recordings (see Figure 1). Additionally, the text associated with each story provides a semantic database against which stories from one newscast can be compared with those from another in building the synthetic newscast tracking a particular user's preferences.

SNR relies on two sources of information, both delivered by satellite feeds from the ABC Radio Network system. The first is hourly five-minute newscasts; all the audio heard through SNR is contained in these newscasts. The second is a text and audio "sound bite" product delivered to ABC affiliate stations, called Newscalls. SNR uses the text portion of Newscalls for its semantic database.

The user hears newscasts by means of either a telephone-based or GUI browsing application, making choices by mouse or Touch-Tone** keys. Both interfaces also provide access to the synthetic newscast. Additionally, audio updates on stories of special interest may be played as they are received either at the user's desktop computer or via a portable voice pager device.

In this paper, we first discuss the segmentation and classification of the audio newscast, using phrase detection and speech recognition techniques. We then describe text-based techniques for clustering the stories of the day; clustering underlies both the segmentation of the audio newscast as well as tracking stories across newscasts. We then discuss the synthetic audio newscast and, subsequently, the user interfaces that interact with it and user evaluations of the interfaces.

The reader should note that although we describe a method for improving speech recognition by integrating additional knowledge, our primary goal was to explore new user interface paradigms. The performance of the system would certainly improve by taking a more disciplined information-retrieval approach to its design. The algorithms we developed are tuned to the specific data set we used for interface evaluations, and we cannot comment on how generalizable they may be.

Intonational phrase classifier

SNR uses a multistage approach to automatically identify story boundaries in audio news broadcasts. The first part, an *intonational phrase* classifier, identifies *possible* story boundaries by breaking the news broadcast into a series of utterances. An intonational phrase expresses at most a single thought or idea. A story consists of multiple phrases; the story boundaries coincide with the boundaries of its first and last phrase. Phrase classifiers attempt to identify transitions from one basic unit of speech to the next. Indicators may include changes in fundamental frequency (perceived as pitch), changes in duration (perceived as the shortening or lengthening of syllables or words), changes in intensity (perceived as loudness), alterations of vocalization with silence (perceived as pausing), changes in voice quality of various kinds, and sometimes changes of turn.[5] Previous studies have shown good results in the automatic identification of intonational phrases.[6]

In addition to their use in identifying story boundaries, intonational phrases as a basic unit of structure present several other benefits in the context of this application. An intonational phrase classifier is a useful front end for an automatic speech recognition (ASR) engine, and will hopefully improve recognition performance by providing salient acoustic and semantic boundaries. Intonational phrases also serve as a basic unit for audio browsing applications.[6]

Phrase detection. Simple pause detection was inadequate because it is difficult to accurately combine the short identified phrases into longer segments, due to a minimum phrase length restriction. In addition to pauses, we use localized rising or falling intonation based on rules identified by Pierrehumbert and Hirschberg.[7] We attempted to create a classifier based on features successfully used in such systems in the past.[6]

Analysis is performed using Entropics Waves software to derive F0 (fundamental frequency) and PVOICE (probability of voicing, for pauses). Two additional derived features are window averages—F0_WIN_MEAN and PVOICE_WIN_MEAN. The windows are 120 ms long (12 frames) and are right-aligned with the current sample, so that only past information was examined. The final rules for finding phrase boundaries follow:

- Flag a localized frequency jump if (F0 $>$ 1.8 $*$ F0_WIN_MEAN) for rising intonation or (F0 $<$ 0.5 $*$ F0_WIN_MEAN) for falling intonation
- Declare a phrase boundary if PVOICE_WIN_MEAN falls below 0.10 within 100 ms (10 frames) of a frequency jump
- Declare a phrase boundary if (PVOICE $=$ 0) continuously for 350 ms (35 frames)

We arrived at the particular parameters chosen here, and elsewhere in this paper, by optimizing performance for the particular data sets used in this project. Later the system imposes one additional constraint: the classifier must produce phrases longer than four seconds. Since phrases will be transcribed by ASR and then classified semantically, shorter phrases do not have sufficient information for reliable classification. Phrases with the shortest pause-defined boundaries are combined first, followed by those with the shortest pauses between pitch-defined boundaries.

The ABC news broadcasts occur once an hour, last for five minutes, and are surrounded by silence. We defined several fixed-phrase boundaries within each broadcast. To find the beginning of each news segment, a simple algorithm averages the samples of a period of known silence to find the ambient noise level, then searches for a 20 percent deviation from that level to signify the start of the broadcast. Two commercial segments occur at relatively constant offsets and lengths from the start of the news. The system automatically removes these segments from the file and declares phrase boundaries at their endpoints.

Performance. We analyzed a single day (March 24, 1999) of news broadcasts from 8:00 AM to 6:00 PM.

The actual boundaries were hand-labeled and are accurate to about 100 ms. We did not include the 2:00 PM broadcast or the first segment (120 seconds, up until the first commercial break) of the 3:00 PM to 6:00 PM broadcasts because each consisted only of a single story. These stories were unusually long because they covered the start of the NATO air offensive against Yugoslavia. The total data set includes 26 minutes of news and 49 actual internal story boundaries representing 65 stories. The remaining boundaries coincided with the starts and ends of commercials and the broadcasts themselves. Figure 2 summarizes the results, and Figure 3 shows a histogram of phrase lengths. The bins are one second each beginning with the value below each bin.

For every actual internal story boundary the system found 3.8 phrase boundaries. The average (true) story length was 24 seconds, and the average identified phrase length was 8.3 seconds. The average error for missed boundaries, 3.05 seconds, is the average amount of time between an actual story boundary that was missed by the system and the closest phrase boundary automatically identified by the system. An analysis of the missed story boundaries shows that some did not have noticeable phrase breaks as the announcer maintained speech momentum through the end of one story into the beginning of the next. More often, however, phrase breaks were initially identified but later lost due to the combination of phrases to meet the minimum phrase length. This is shown by the fact that the average error for missed boundaries, 3.05 seconds, is less than the minimum phrase length of four seconds. Overall, the minimum phrase length restriction was not very detrimental because the content of a story as a whole can still be understood without the first or last few seconds of the story.

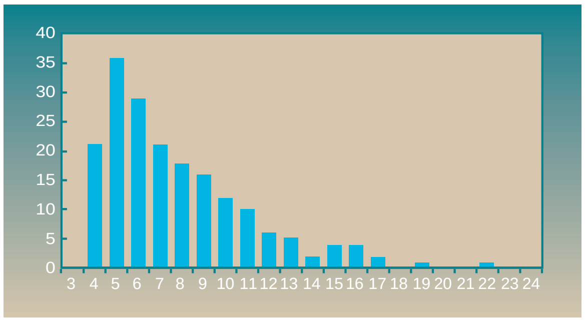
Text story group knowledge

The phrase classifier provides many possible story boundaries. In order to better discriminate among these possibilities, and eventually match stories from one broadcast to another, the system needs to derive semantic information from the news audio. Synthetic News Radio uses automatic speech recognition to obtain this semantic information, and we will describe how an additional source of contextual knowledge is ultimately needed to help combine intonational phrases into full stories and to match stories to one another.

Figure 2 Analysis of the intonational phrase classifier

TOTAL PHRASES IDENTIFIED :	188
AVERAGE PHRASE LENGTH:	8.30 SECONDS
MINIMUM PHRASE LENGTH:	4.02 SECONDS
MAXIMUM PHRASE LENGTH:	22.39 SECONDS
STORY BOUNDARIES FOUND EXACTLY:	31/49 = 63%
AVERAGE ERROR FOR MISSED BOUNDARIES:	3.05 SECONDS
AVERAGE ERROR FOR ALL BOUNDARIES:	1.12 SECONDS

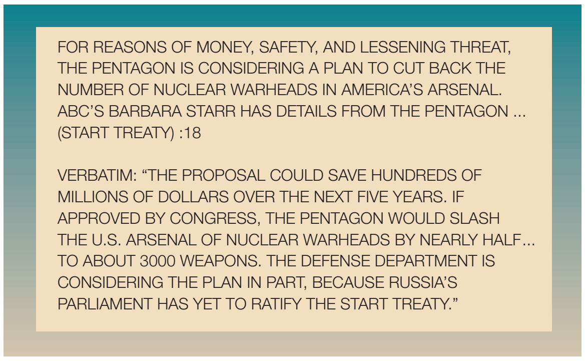
Figure 3 Phrase lengths separated into one-second bins



This section describes the use of text news stories, the Newscalls mentioned earlier, to provide a basis of knowledge about the most prominent news stories over a short period of time—up to one day. An ABC satellite news feed provides Newscalls to affiliate radio stations to allow them to assemble their own newscasts with local announcers. A Newscall consists of an audio portion, typically a sound bite from a prominent person in the news or a local reporter, and a text portion, which gives a brief contextual description for the sound bite along with a transcription of the audio quote, if applicable. Figure 4 shows an example of the content of the text portion of a Newscall.

The system discards the audio portions but uses the text segments to define the important news stories specific to the present time. In the future, when we refer to a Newscall, we are referring only to the text portion. An algorithm clusters these Newscalls, based on their semantic similarity, into larger groups, each representing a single major news story. The system will later use these clusters to help group intonational

Figure 4 Text portion of a Newscall



FOR REASONS OF MONEY, SAFETY, AND LESSENING THREAT, THE PENTAGON IS CONSIDERING A PLAN TO CUT BACK THE NUMBER OF NUCLEAR WARHEADS IN AMERICA'S ARSENAL. ABC'S BARBARA STARR HAS DETAILS FROM THE PENTAGON ... (START TREATY) :18

VERBATIM: "THE PROPOSAL COULD SAVE HUNDREDS OF MILLIONS OF DOLLARS OVER THE NEXT FIVE YEARS. IF APPROVED BY CONGRESS, THE PENTAGON WOULD SLASH THE U.S. ARSENAL OF NUCLEAR WARHEADS BY NEARLY HALF... TO ABOUT 3000 WEAPONS. THE DEFENSE DEPARTMENT IS CONSIDERING THE PLAN IN PART, BECAUSE RUSSIA'S PARLIAMENT HAS YET TO RATIFY THE START TREATY."

phrase transcripts into stories as well as match stories across multiple broadcasts. SNR uses Latent Semantic Indexing to cluster related Newscall files into stories.

Latent Semantic Indexing. Latent Semantic Indexing (LSI), [8] or Latent Semantic Analysis (LSA), is a variant of earlier information retrieval techniques such as SMART.[9] It differs from those methods in that it also extracts information about the natural co-occurrences of words. It does not use any information about transition probabilities from word to word, but instead uses a singular value decomposition (SVD) to deduce relationships between words that appear in large natural passages. Documents are represented with the same term vectors and weighted in similar ways; however, LSI (detailed below) uses an entire collection of domain representational texts to form a large term-document matrix, which will be used to transform the term vector representation for each query document.

Stemming and removal steps. When analyzing Newscalls to create an index, the first step of the process stems words to remove morphological variations. Plural endings, adverb endings, verb forms, etc., do not significantly alter the meaning of a word and are removed so that all forms of a word are seen as identical to the system. We use a C implementation by Frakes and Cox of the Porter stemming algorithm.[10]

The next step of the process removes words that appear on a list of 578 common words. These words appear often and do not carry a significant amount of semantic meaning. Some examples include: *also*, *after*, *and*, *did*, *do*, and *each*. We used the list of common words implemented in the SMART system, along with some additions specific to Newscalls, such as *ABC* and *verbatim*.

Creating the index. A program automatically generates a new Latent Semantic Index once an hour throughout the day. It uses all the Newscalls received so far for the current day. After words have been stemmed and the common ones removed, the system converts each text Newscall to a term vector, and combines them into a weighted term-document matrix according to the standard LSI technique,[8] using the log-entropy weighting scheme[11] for local and global weights, respectively. This entropy scheme gives less weight to terms that occur frequently or are equally distributed across a large number of documents and gives more weight to terms that can better discriminate between documents. The system then performs an SVD on the term-document matrix and keeps the vectors associated with the 20 largest singular values as the index. The index represents the semantic relationships found in the set of documents.

To compare two passages of text, each one is represented as a term vector and weighted as before, then transformed to a 1×20 reduced vector through multiplication with the index. The cosine (or dot product) of the two vectors, scaled to unit length, becomes a measure of their semantic similarity, with higher values meaning more similarity.

Text story clustering. Every hour, immediately after the system creates an index, it executes an algorithm to cluster the Newscalls into larger groups representing prevalent topics in the news. A few major stories will have a large number of Newscalls representing them, whereas others will have just a few. An examination of the Newscalls showed a very high degree of similarity between Newscalls about the same stories. Many would have topic descriptions that were almost verbatim duplicates. This similarity led to a two-stage approach consisting of (1) word matching to create initial Newscall clusters, and (2) using LSI to match the remaining unassigned passages. The algorithm uses the following technique:

- Group together Newscalls that have more than 15 percent identical words between them, counting only stemmed words that appear in the index.
- For each remaining Newscall, determine with which already assigned Newscall the remaining one has the highest LSI correlation, and assign it to the

same cluster, as long as the highest cosine similarity value is >0.7.
- Any Newscalls left at this point will not receive a cluster assignment and will not be used in future stages of processing.

This method precludes the existence of a single-story cluster; however, singleton stories are not likely to appear in the audio news broadcast.

Performance. The evaluation data consist of Newscalls collected for one entire day (November 23, 1998). The Newscalls were grouped by one person according to topic similarity into a total of 36 story clusters. The above algorithm was then used to automatically cluster the Newscalls into a total of 23 clusters. Figure 5 shows the hand-labeled data designated as "human" next to the assignments by algorithm.

Each pair of bars in the figure represents the assignments of the exact same Newscalls, not just the same number of Newscalls. Bars with horizontal lines represent more than one cluster. The figure shows that, in some cases, the algorithm combined several clusters defined by the hand-labeler into a single large cluster. For example, leftmost bars show that the person placed 83 Newscalls into four different clusters and that the algorithm placed 82 of those same 83 Newscalls into a single cluster. The extra Newscall (which represented a single story cluster related to Kenneth Starr) was left unclassified by the algorithm. The remaining three clusters grouped together contained stories about:

- The response of the Clinton administration to Iraq's refusal to allow access to United Nations weapons inspectors
- Clinton's visit to Guam for a memorial for Americans killed in World War II
- Discussion of the Clinton impeachment inquiry

However, the algorithm rarely separated Newscalls grouped together by the labeler into different groups. Separation happened only once with the single Newscall that was left unclassified. These results mean that the system leans toward identifying two unrelated stories as related rather than failing to match two stories that are indeed the same.

Further tests, for matching stories across multiple broadcasts, showed that disproportionately large clusters still tend to produce high correlations with unrelated stories, so the final version of the clustering algorithm limits the number of Newscalls per cluster to 25 and discards the remaining ones. The Newscalls are generally evenly distributed so that if a large cluster actually contains more than one story, eliminating extra Newscalls from it does not eliminate an entire story.

Figure 5 Results from the Newscall clustering algorithm

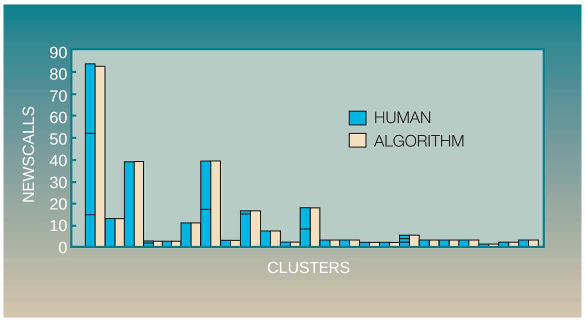


Figure 6 Example output from speech recognizer

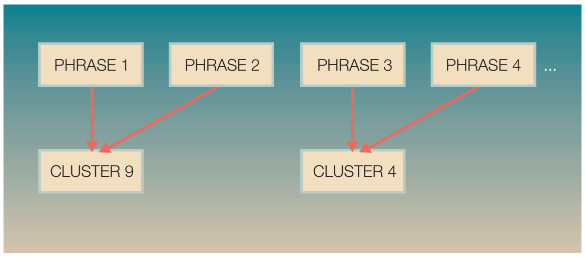
Speech recognizer: some of the NATO pilots found in the service life in the carrier Roosevelt with a radical of working in the field box

Actual transcription: Some of the NATO pilots pounding the Serbs fly from the carrier Roosevelt in the Adriatic. They call it working the kill box.

Finding story boundaries

The last sections described the process of identifying intonational phrases and of clustering text Newscalls by creating semantic vector representations for each passage of text. This section discusses the final step in finding story boundaries—using the information provided by the Newscall clusters to properly combine the intonational phrases into full stories. This phase uses automatic speech recognition (ASR) software to transcribe each intonational phrase, providing semantic information about the broadcast. We use the LSI to compare the semantic information in the broadcast to the information in the various story clusters. Vector representations of the phrase transcripts, created using the index, are matched to story

Figure 7 Combining intonational phrases to create stories



clusters to help determine which phrases are likely about the same story.

Automatic speech recognition. Synthetic News Radio uses a commercially available large-vocabulary continuous speech recognition engine. Figure 6 shows a sample transcription from an intonational phrase about the air strikes in Yugoslavia, generated by this product.

The recognizer does not offer much additional information. It does not return any measure of confidence, nor does it give any indication if it fails to recognize anything from certain portions of the audio. New proper nouns in the news, especially names, sometimes contributed significant errors because they could not be recognized correctly. A study of ASR performance on broadcast news agrees with our experience, showing an average of 67 percent words correctly recognized.[3] Using semantic information from the transcripts alone is not enough to accurately find story boundaries, so the system uses additional information in the form of the text story clusters. Although there is currently a strong research effort funded by DARPA (Defense Advanced Research Projects Agency) and NIST (National Institute of Standards and Technology) to improve recognition accuracy for broadcast news, high accuracies are not really needed for this project. The goal is not to obtain an accurate transcript, but simply to extract enough semantic information to create a representation that the system can use to find relevant stories.

Grouping intonational phrases. Every hour the system generates a new LSI, and the most recent index is used to transform each intonational phrase transcript into a representational vector. The system then determines which Newscall cluster has the highest correlation with each phrase. The hypothesis is that sequential phrases about the same story will have high correlations with the same story cluster. A story boundary then occurs when the phrase correlations switch from one story cluster to another. Figure 7 depicts this process.

The actual results are not as clean as shown in this figure, so the sequencing algorithm described in the next subsection attempts to locate the best matches and smooth out errors.

Sequencing algorithm. The sequencing algorithm generates a vector for each transcript according to the process outlined earlier, using only words that appear in the index. This action helps eliminate recognition errors, because words that do not appear in the index were most likely transcribed incorrectly. In one specific exception it searches for a story match based on keywords. It performs this search for Wall Street stock market updates, because results showed that this story almost always occurs in the broadcast, but generally has very few, if any, Newscalls associated with it.

The algorithm uses both the similarity cosine and also the absolute magnitude of the vector (measured before it is scaled to unit length for the cosine determination). Vectors with low magnitudes generally do not have enough semantic information to make accurate matches. For each intonational phrase, the system finds the story cluster that has the highest correlation (cosine), and then follows a series of steps to smooth out inconsistencies in the cluster assignments. For example, if phrases four and six correlate highly with cluster six, and phrase five has a poor correlation with cluster nine, or perhaps a low magnitude, then the algorithm will combine phrases four, five, and six into a single story.

Performance. Evaluation for this phase uses the same audio data and the corresponding Newscall clusters, described earlier. The segments omitted before are omitted again here. The data consist of 26 minutes of news, which includes 67 actual stories (identified by hand-labeling) and 188 phrases identified by the phrase classifier. Figure 8 summarizes the results.

We evaluated performance only on *internal* story boundaries, which do not include fixed boundaries defined by the beginning and end of the broadcast, and the commercial breaks. Perceived performance is therefore somewhat better than that shown here because the fixed boundaries are always accurate. The average error for missed boundaries is about one second higher than the same error for the intonational phrase classifier. This difference shows that the algorithm occasionally groups together transcripts across a real story boundary, but the large ratio of found boundaries to actual boundaries (1.5:1) shows that it was more likely to break a real story into multiple segments. An analysis of errors in grouping transcripts across a real boundary shows that the boundary was often missed by a single intonational phrase, and that this phrase contained very little story-related semantic information, such as a speaker and location identification message. Errors in breaking stories into smaller segments occurred either because of a lack of information in the ASR transcription (sometimes due to extended periods of live, natural speech, such as a witness account) or because of poor correlation with the appropriate Newscall cluster.

Synthetic newscasts

A synthetic news broadcast is a custom compilation of stories drawn from recent news broadcasts. The selections made by the user (described in the following section) affect which stories the system selects for the synthetic broadcast, as well as the presentation order of those stories. A user may listen to one or more early broadcasts in the morning, giving the system feedback about topic preferences, and then receive a full synthetic newscast near the end of the day, tailored to the preferences defined that morning or even to preferences defined in previous days. A synthetic newscast is derived from the user query file (which indicates what stories to follow or ignore) and a sequence of actual newscasts during the day.

Matching stories across multiple broadcasts. Matching a story from one broadcast to that of another broadcast works in much the same way as the sequencing algorithm. The transcripts of the selected stories are saved in the user query file.

Every time the system tries to make a match, it creates a new semantic vector representation for the story using the most recently generated index. Next, it calculates cosine similarities for each defined story in a segmented broadcast. It matches the query to the story with the highest correlation, as long as that value is above a threshold of 0.7. One exception to this method occurs again with the stock market updates for which the system matches keywords, because they are too short.

Figure 8 Sequencing algorithm results for finding story boundaries

TOTAL HAND-LABELED (HL) STORIES:	67
TOTAL SYSTEM STORIES:	90
TOTAL HL INTERNAL BOUNDARIES:	49
TOTAL SYSTEM INTERNAL BOUNDARIES:	74
ACCURATE SYSTEM BOUNDARIES:	26 / 49
AVERAGE ERROR FOR MISSED BOUNDARIES:	4.0 SECONDS

A current limitation to this technique is its inability to detect subtle topic differences. It can determine whether two stories generally convey information about the same topic, but it cannot determine whether a story has changed or whether one contains any additional information that a previous version did not have.

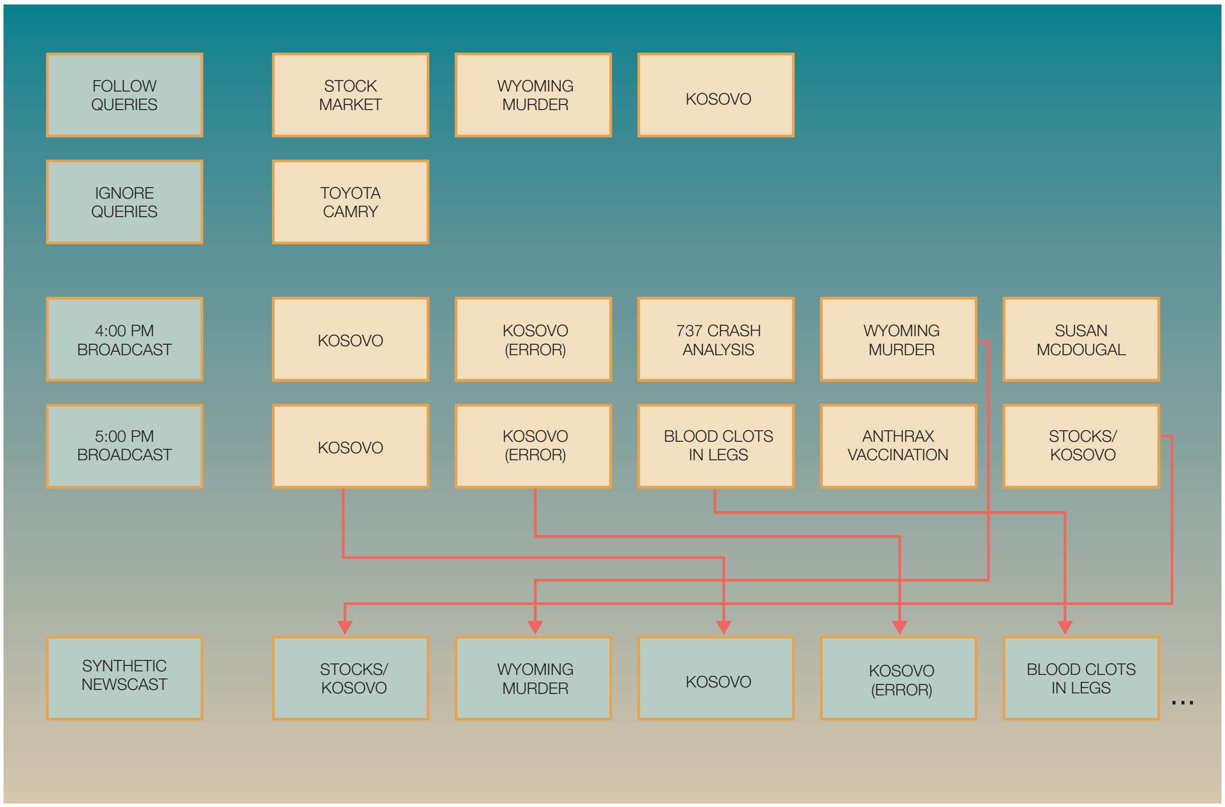
Construction. A synthetic newscast places the most interesting stories for the user at the beginning of the newscast, according to their defined priority, and then adds other stories to the end. Figure 9 illustrates the construction of a synthetic newscast.

In this example, the "stock market" story was found in the 5:00 PM broadcast, the "Wyoming murder" story in the one at 4:00 PM, and the "Kosovo" story again in the one at 5:00 PM. The stock market story has a combination boundary error because it includes a Kosovo recap section at the end. The broadcast Kosovo stories also have segmentation errors because a small section at the end was not included with the main story group. This small segmentation error appears next in the synthetic newscast as a result of bad correlation, followed by new stories from the 5:00 PM and then the 4:00 PM broadcasts.

Construction follows this general procedure:

- Starting with the highest priority follow story (the first one listed in the user query file), search for a match for each story. If matches are found in multiple broadcasts, then select the most recent one.
- Search for stories that match entries in the ignore list, and remove each of those from consideration.
- Place the remaining unmatched stories at the end of the synthetic broadcast, also adding the most recent ones first.
- Place the final "This is ABC news" blurb at the end.

Figure 9 A synthetic newscast



Typically the system will automatically generate custom newscasts one to three times per day.

Evaluations of the story selection process showed that users were sometimes confused about how to follow or ignore stories with inaccurate boundary definitions. This confusion happened mostly when long stories were split into two or more segments, but also occurred when more than one story was combined into a single segment. For the first case, selecting all story segments about a topic of interest generally produced the best matching results in the synthetic newscast. Matching for the second case was somewhat unpredictable due to the combined information about more than one story.

Listening to the synthetic newscast. The synthetic newscast can be heard using either the graphical or telephone-based interface described later. When using the graphical desktop interface, most people wanted a visual representation of how the system created the newscast. They requested that the application display a few of the most discriminating words used to make the match for the current story, perhaps highlighting the parts of the query transcript that the system deemed most relevant to the matched story. When listening to the newscast over the phone, presentation of this ancillary information was not deemed as important because of the already restricted communicative capabilities of the medium. Segmenting errors that added or removed several seconds of a story near the boundary proved to be annoying to users, even if they understood the main point.

Matching important stories. New ABC radio broadcasts arrive once every hour, and the system searches each of these broadcasts for matches to important query stories. It cannot really determine whether the latest version of the story contains new information, or whether it has changed significantly since the last one received, so it always forwards matches from new broadcasts.

User interface design

Although we have described techniques for segmenting and classifying audio newscasts and presented the synthetic newscast, continually updated to reflect a user's interests, we have not yet described the user interfaces whereby the user hears and interacts with the news. This section describes the design criteria for such user interfaces, and the following two sections describe a graphical and a telephone-based user interface.

The user interfaces must allow control over playback of the newscast audio, as well as a means of marking stories for following. Although text may be included as a navigational aid, it is not required for indicating a topic of interest. The interface must support the fact that newscasts are updated hourly.

Because each audio newscast is segmented and classified, the interfaces should support playback at the story level, rather than simple time offset into the audio file. The user should be able to jump ahead to the next story if the current one is uninteresting or jump back to a previous story. In addition, it is desirable to support scan features to allow rapid preview of all the stories.

As the news changes constantly, or at least hourly for this application, it is useful to support a variety of access methods and asynchronous delivery. We desire to support news retrieval both in conventional computer usage situations as well as for users who are mobile, driving, etc.

The underlying text-based semantic representation is to be hidden from the user. All the user needs to do is listen to the news, and optionally express interest in following or killing a story just by taking an action while hearing that story.

Many interface features were introduced, changed, or deleted through user evaluation sessions. These changes are noted along with the discussions on that particular interface element. A total of seven people participated in evaluations at one time or another. Each user was asked to participate in the process of selecting stories to follow using the graphical interface, and to then listen to and navigate through a synthetic newscast using either the graphical or phone interface. Their experiences were recorded through a verbal question and answer period.

Visual desktop interface

An X-Window System** application, *xnews*, provides the primary interface for a user sitting at a computer. In a more elaborate version of the system, we anticipate the user will primarily use the visual interface to make initial customization decisions, which future interactions with the system can then build on. Our design focuses on simple but useful interface elements that are easily extensible to other platforms, such as Web pages or mobile devices. Figure 10 shows the main window.

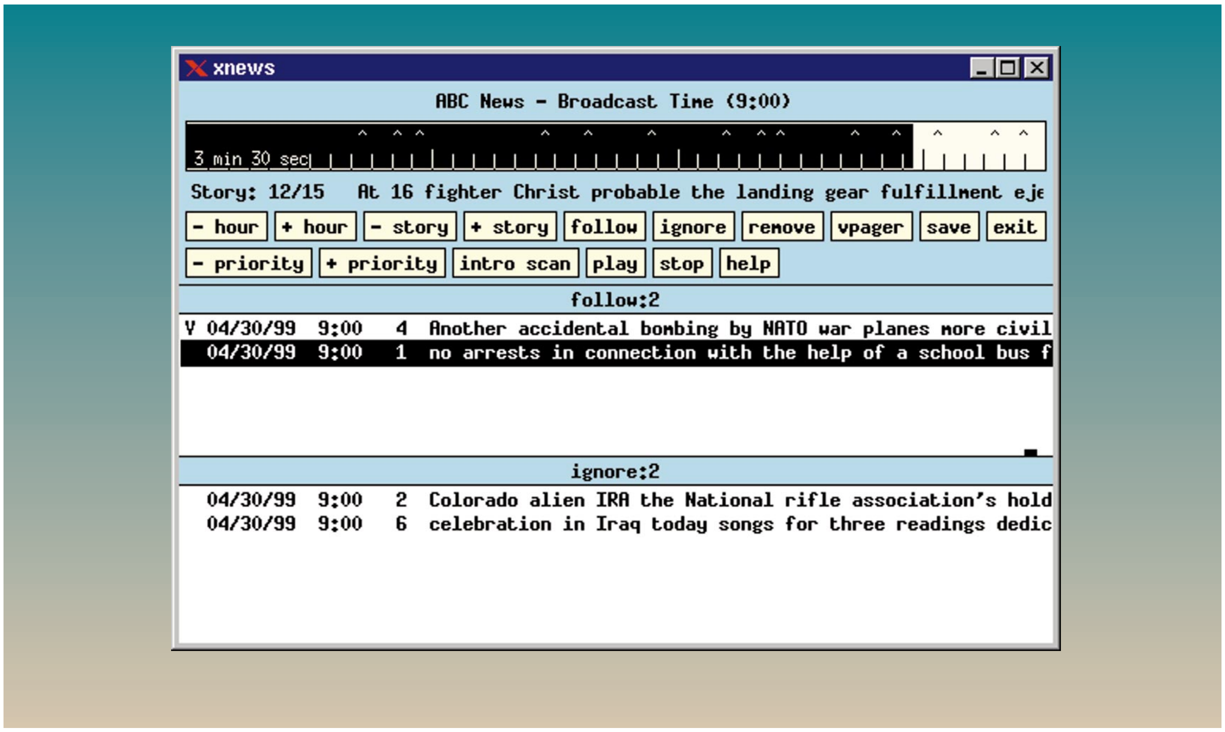
The *xnews* interface allows browsing of multiple hourly broadcasts as well as the user's most recent synthetic broadcast generated by the system. It also lets the user select stories that it will use for future filtering and broadcast construction decisions.

The audio of the current broadcast is the main focus of the application window, as shown in the *SoundViewer* widget of Figure 11. This widget controls playback of the audio and has many built-in features. Via mouse control (clicking and dragging in the widget itself) the user can start and stop playback and jump instantaneously to any point in the audio file. The widget also supports an audio time compression algorithm, allowing playback at higher speeds than normal without changing the pitch, but gradually degrading the comprehensibility of the audio.

Users expressed a desire for a visual representation of the length of each story, so we added widget annotations. Carat marks indicate the positions of the story boundaries within each newscast.

The SoundViewer widget can be quickly moved to the annotated story boundaries by using the ± story buttons. Along with the relative time position in the broadcast shown by the widget, two other indicators describe the current story. First, a field displays the current story index number along with the total number of stories for the current broadcast. Second, a portion of the transcription from the ASR engine for that particular story is displayed, as seen just below the SoundViewer widget in Figure 11. This text dis-

Figure 10 Graphical *xnews* application



```
ABC News - Broadcast Time (9:00)
Story: 12/15   At 16 fighter Christ probable the landing gear fulfillment eje
follow:2
V 04/30/99   9:00   4  Another accidental bombing by NATO war planes more civil
  04/30/99   9:00   1  no arrests in connection with the help of a school bus f
ignore:2
  04/30/99   9:00   2  Colorado alien IRA the National rifle association's hold
  04/30/99   9:00   6  celebration in Iraq today songs for three readings dedic
```

play is similar to the NewsTime project,[1] except that this project used text from closed caption encoding.

Another technique, the *intro scan*, is similar to the familiar radio function that scans multiple stations. It plays the first five (user-definable) seconds of each story sequentially. Some users thought the intro scan was a useful tool for quickly browsing and selecting stories; others thought it was easier to simply use the navigation buttons to move from story to story, arguing that the intro scan did not offer enough information to accurately determine the topic of every story.

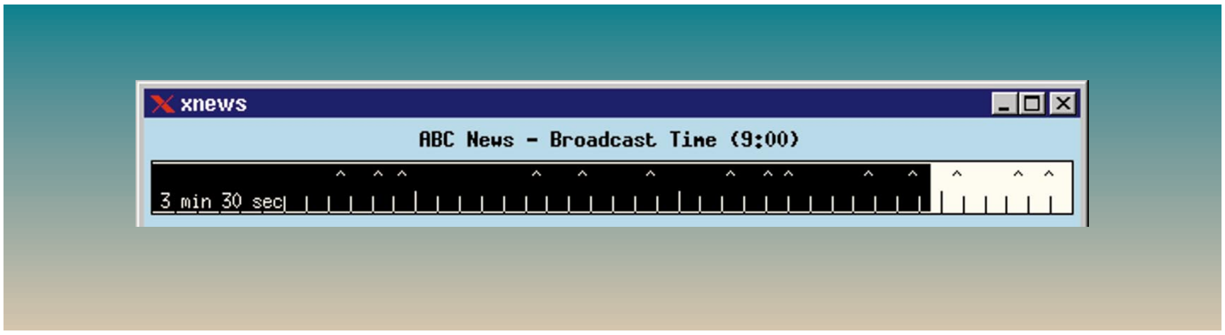
While listening to a broadcast, the user can tell the system to either *follow* or *ignore* the current story in the future. This feature adds a description of the story to the appropriate list in the application window, as shown in Figure 11. A story in the follow list may also have a "V" flag that identifies the story as a particularly important one, and tells the system to attempt immediate delivery of future matching stories.

Stories to be followed in the synthetic newscasts will be presented in the same order in which they appear in the list. Two interface buttons allow the user to increase or decrease the priority of a story once it has been added to the list.

The text descriptions in the lists come from the ASR transcripts. Because the transcripts are error prone and story boundaries inaccurate, the text presented is from the location in the story at which the user selected it while listening. We assume a linear relationship between the number of words and time length of the story. For example, if the user pressed *follow* halfway through the audio of a story, the application would display text from the middle of the transcript for that story.

Listeners were generally able to use the transcription segments in the list to remember which audio stories they selected during a single session; however, if they reviewed their selections the next day, they had a much more difficult time associating the text segment with news topics. Some wanted the ability to review the entire ASR transcript for a story, whereas others wanted to see only a few keywords that accurately summarized the topic.

Figure 11 SoundViewer widget



Telephone user interface

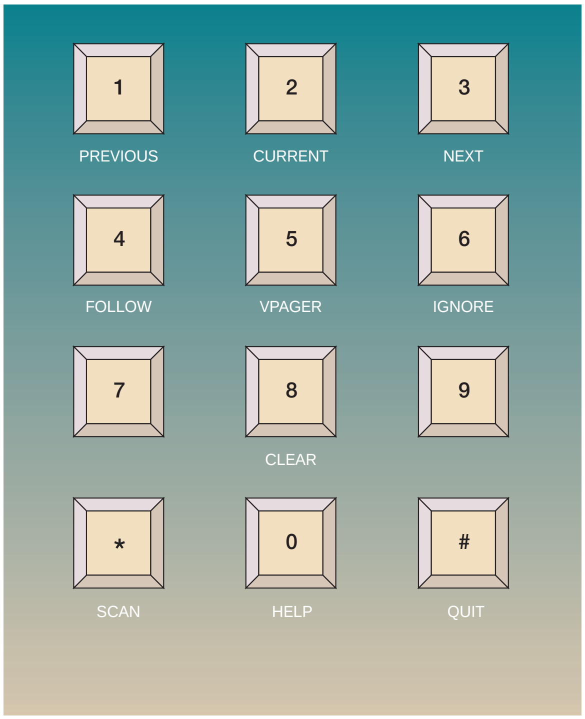
A Touch-Tone-driven phone application, built as part of the Phoneshell[12] system, significantly increases the accessibility of the news. It enables access to audio news in environments far from the standard desktop computer, including ones where the user is mobile, such as walking, riding in a car, or traveling. Figure 12 shows a layout of the telephone interface.

It would, of course, be preferable to deliver commands to the system by voice, at least in the imagined scenario of a single subscriber using the system while driving alone. But note that the only representation of the story is the content of the associated cluster itself. One of the hypotheses of Synthetic News Radio was that we could interact directly with the story audio, which would mean a limited recognizer vocabulary such as "next story" and "follow this" rather than sophisticated queries based on keywords (which would likely be difficult to recognize).

Story navigation is similar to the graphical interface, but without visual feedback. The system does not give any explicit audio feedback as the user switches from story to story, because the change in news audio already indicates a successful navigation operation. However, it does notify the user if navigation beyond the first or last story is attempted. Scan mode plays introductory snippets from each story. In a more advanced version of the system, we can imagine supporting hyperlinked audio. The system could play headline summaries of the main stories and allow the user to hear an entire story only if the headline sounds interesting.

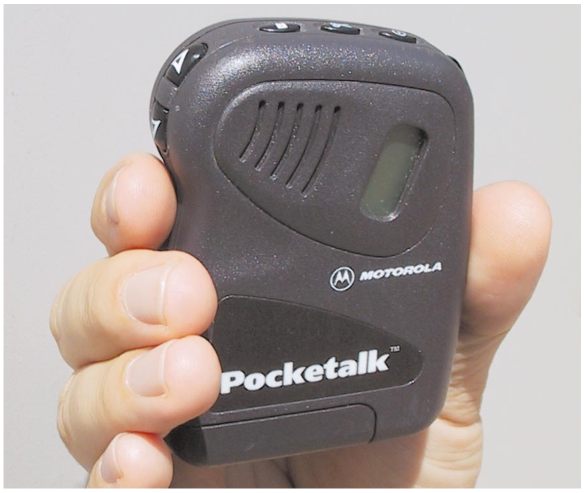
Although the phone interface is meant primarily as a browsing tool for customized newscasts, users can also make additional queries. While they listen to the audio, Phoneshell uses the same selection technique as *xnews*, allowing users to identify stories to follow, ignore, or send as alerts. The interface does not support story priority definition as *xnews* does.

Figure 12 Command-to-key mapping for the telephone interface



We have experimented with a variety of auditory feedback cues, and thus allow per user customiza-

Figure 13 Voice pager



tion with a configuration file. Feedback to the various selection operations may be text-to-speech or canned or user-specified sounds. Most users liked the short audio cues the best. Interestingly, however, one user thought any sort of audio response whatsoever was enough to interrupt the train of thought, so she opted for no feedback to story selections over the phone.

The interfaces above described ways to indicate interest and disinterest in particular stories heard by the users. The applications save the selections in a user query file, which is unique to each user of the system. Stories selected one day automatically remain in the file in the future until they are explicitly deleted. This retention works well for interest in longer-term stories that may continue for days or even weeks. The system will continue to search for matches to the descriptions it has in the query file.

Mobile device interfaces

We also experimented with several techniques to deliver news to a mobile user in a time-sensitive manner. When an hourly newscast contains a story that a user is following with particular interest, that story can be extracted and delivered asynchronously. We use the UNIX** "finger" utility, which monitors keyboard activity, to determine whether a user is sitting at his or her desk. The updated story can play over the local area network (LAN) through the user's desktop computer either immediately or when an inactive user returns.

Another asynchronous method forwards stories to a voice pager when the user is away. A voice pager is a portable electronic device that receives 60-second voice clips as pages. Figure 13 pictures one such device.

This delivery method will become more appropriate as technologies converge and more functionality is added to a single mobile device.

Finally, we implemented an interface specifically for text financial news, organized around a user's personal portfolio. Relevant financial headlines are automatically sent to the user's pager or messaging-enabled cellular phone. The user can then place a call over the phone and hear an entire synthesized story, and also browse through old stories.

Conclusions

We developed a system that automatically searches for story boundaries in radio news broadcasts, and then creates an annotation file useful for browsing and filtering. Automatically generating structure for news audio enabled the creation of new interfaces that support nonlinear navigation, and generating semantic story representations created a mechanism for assembling a synthetic newscast from audio-based queries. The intonational phrase classifier performed reasonably well and correctly identified most of the story boundaries. When it missed a boundary, it still found a natural-sounding break in the speech stream. The Newscall clustering algorithm identified groups of stories similar to those defined by a real person; however, it sometimes spread the topic definition by combining too many topics into a single group. The algorithmic parameters we chose may not generalize well to broader data sets.

User interfaces demonstrated effective use of the annotated story boundaries as a browsing and navigation aid. The audio query selections were successfully applied to future radio broadcasts to create synthetic newscasts; however, users requested more feedback as to exactly how the system matched the selected stories to the audio-based queries. Delivering audio news directly to mobile devices should prove useful as wireless bandwidth increases and mobile devices become audio-enabled.

Acknowledgments

We thank ABC News Radio for the use of their audio and text news feeds, and the News in the Future research consortium. Mark Ackerman wrote the SoundViewer widget. We would also like to thank the many anonymous readers who helped improve this paper.

**Trademark or registered trademark of Fred D. Hinger, X Consortium, Inc., or The Open Group.

Cited references

1. C. Horner, *NewsTime: A Graphical User Interface to Audio News*, S.M. thesis, MIT, Cambridge, MA (1991).
2. A. Hauptmann and M. Witbrock, "Informedia: News-on-Demand Multimedia Information Acquisition and Retrieval," *Intelligent Multimedia Information Retrieval*, M. T. Maybury, Editor, AAAI Press, Menlo Park, CA (1997), pp. 213–239.
3. S. Whittaker, J. Hirschberg, J. Choi, D. Hindle, F. Pereira, and A. Singhal, "SCAN: Designing and Evaluating User Interfaces to Support Retrieval from Speech Archives," *Proceedings of ACM SIGIR '99* (1999), pp. 26–33.
4. M. Viswanathan, H. S. M. Beigi, S. Dharanipragada, and A. Tritschler, "Retrieval from Spoken Documents Using Content and Speaker Information," *Proceedings of the Fifth International Conference on Document Analysis and Recognition* (1998), pp. 567–572.
5. W. Chafe, "Intonation Units," *Discourse, Consciousness, and Time*, The University of Chicago Press, Chicago (1994), pp. 53–71.
6. J. Hirschberg and C. Nakatani, "Using Machine Learning to Identify Intonational Segments," *Proceedings of AAAI Spring Symposium on Applying Machine Learning to Discourse Processing*, Palo Alto, CA (March 1998).
7. J. Pierrehumbert and J. Hirschberg, "The Meaning of Intonational Contours in the Interpretation of Discourse," *Intentions in Communication*, P. R. Cohen, J. Morgan, and M. E. Pollack, Editors, MIT Press, Cambridge, MA (1990), pp. 271–311.
8. T. K. Landauer, D. Laham, and P. W. Foltz, "Learning Human-Like Knowledge by Singular Value Decomposition: A Progress Report," *Advances in Neural Information Processing Systems 10*, M. I. Jordan, M. J. Kearns, and S. A. Solla, Editors, MIT Press, Cambridge, MA (1998), pp. 45–51.
9. *The SMART Retrieval System—Experiments in Automatic Document Retrieval*, G. Salton, Editor, Prentice-Hall, Inc., Englewood Cliffs, NJ (1971).
10. M. F. Porter, "An Algorithm for Suffix Stripping," *Program-Automated Library and Information Systems* **14**, No. 3, 130–137 (July 1980).
11. S. Dumais, *Enhancing Performance in Latent Semantic Indexing (LSI) Retrieval*, Technical Report, Bellcore (now Telcordia Technologies), Morristown, NJ (September 1992).
12. C. Schmandt, "Phoneshell: The Telephone as a Computer Terminal," *Proceedings of ACM Multimedia '93* (1993), pp. 373–382.

Accepted for publication April 20, 2000.

Keith Emnett *MIT Media Laboratory, 20 Ames Street, Cambridge, Massachusetts 02139-4307 (electronic mail: emnett@media.mit.edu).* Mr. Emnett received his M.S. degree from MIT, where he was a research assistant in the Speech Interface Group at the Media Laboratory. There he developed personalized news systems and interfaces to mobile communication devices. He also worked on interfaces for the "Mutually Immersive Robotic Telepresence" project at Compaq Western Research Laboratory. Earlier he received his B.S. degree in electrical engineering from Oklahoma State University, where his Honor's thesis was the "Audio Compass," after which he designed hardware digital signal processing systems at Texas Instruments. He is currently employed at a small Silicon Valley startup company.

Chris Schmandt *MIT Media Laboratory, 20 Ames Street, Cambridge, Massachusetts 02139-4307 (electronic mail: geek@media.mit.edu).* Mr. Schmandt received the B.S. and M.S. degrees from MIT, where he has been building speech systems since 1979. He is the director of the Speech Interface Group at the Media Laboratory, a position he has held since the creation of the laboratory. Before that he worked on speech applications research at the Architecture Machine Group, including the "Put That There" and "Phone Slave" projects, as well as projects in digital video typography and gestural input for stereoscopic video displays. His current research focuses on speech interfaces for highly mobile computers.